# 设计模式笔记总结

## 1.解决问题的两种方法：

分解问题

抽象问题

## 2.设计模式原则

### 面向对象设计的原则一：

依赖倒置原则(DIP)：

高层模块(稳定)不应该依赖于底层模块(变化)，二者都依赖于抽象(稳定)

抽象(稳定)不应该依赖于实现的具体细节(变化)，实现细节应该依赖于抽象(稳定)

### 面向对象设计的原则二：

开放封闭原则(OCP)：

对扩展开放，对更改封闭。

类模块应该是可扩展的，但是不可修改。

### 面向对象设计的原则三：

单一职责原则(SRP)：

一个类仅有一个引起它变化的原因。

变化的方向隐含着类的责任。

### 面向对象设计的原则四：

Liskov替换原则(LSP)

子类必须能够替换它们的基类(IS-A)。

继承表达类型抽象。

### 面向对象设计的原则五：

不应该强迫客户程序依赖它们不用的方法。

接口应该小而完备。

### 面向对象设计的原则六：

优先使用对象组合，而不是继承

类继承通常为“白箱复用”，对象组合通常为“黑箱复用”。

继承在某种程度上破坏了分装性，子类父类耦合度较高。

而对象组合则只要求被组合的对象具有良好定义接口，耦合度低。

### 面向对象设计的原则七：

分装变化点

使用封装来创建对象之间的分界层，让设计者可以在分界一侧进行修改，而不会对另一侧产生不良的影响，从而实现层次间的松耦合。

### 面向对象设计的原则八：

针对接口编程，而不是针对实现编程

不将变量类型声明为某个特定的具体类，而是声明为某个接口。

客户程序无需获知对象的具体类型，只需要知道对象所具有的接口。

减少系统中各部分的依赖关系，从而实现“高内聚，松耦合”的类型设计方案。

设计模式最大的特点就是在稳定和变化点中需求隔离点，从而分离变化点管理变化点。

稳定中有变化

## 3.面向对象中复用的概念：

指的是二进制单位的复用（就是源代码编译，测试之后不再改动，编译单位的复用性），不是指源代码片段中的复用。

## 4.注意点：

1.使用设计模式的时候，需要分析稳定点和变化点，在变化点处使用设计模式。

2.软件的分析不要静态的分析，而是采用动态的分析，从时间轴上去分析，软件的需求是不断的发生变化。

3.多继承使用的正确姿势，一个主类，多个接口。

## 5.设计模式

### 1.模板方法（template method）

可以考虑利用模板方法重写关于http的请求代码

### 2.策略模式